计算机学院专业必修课

计算机组成

多周期处理器
形式建模综合方法

高小鹏

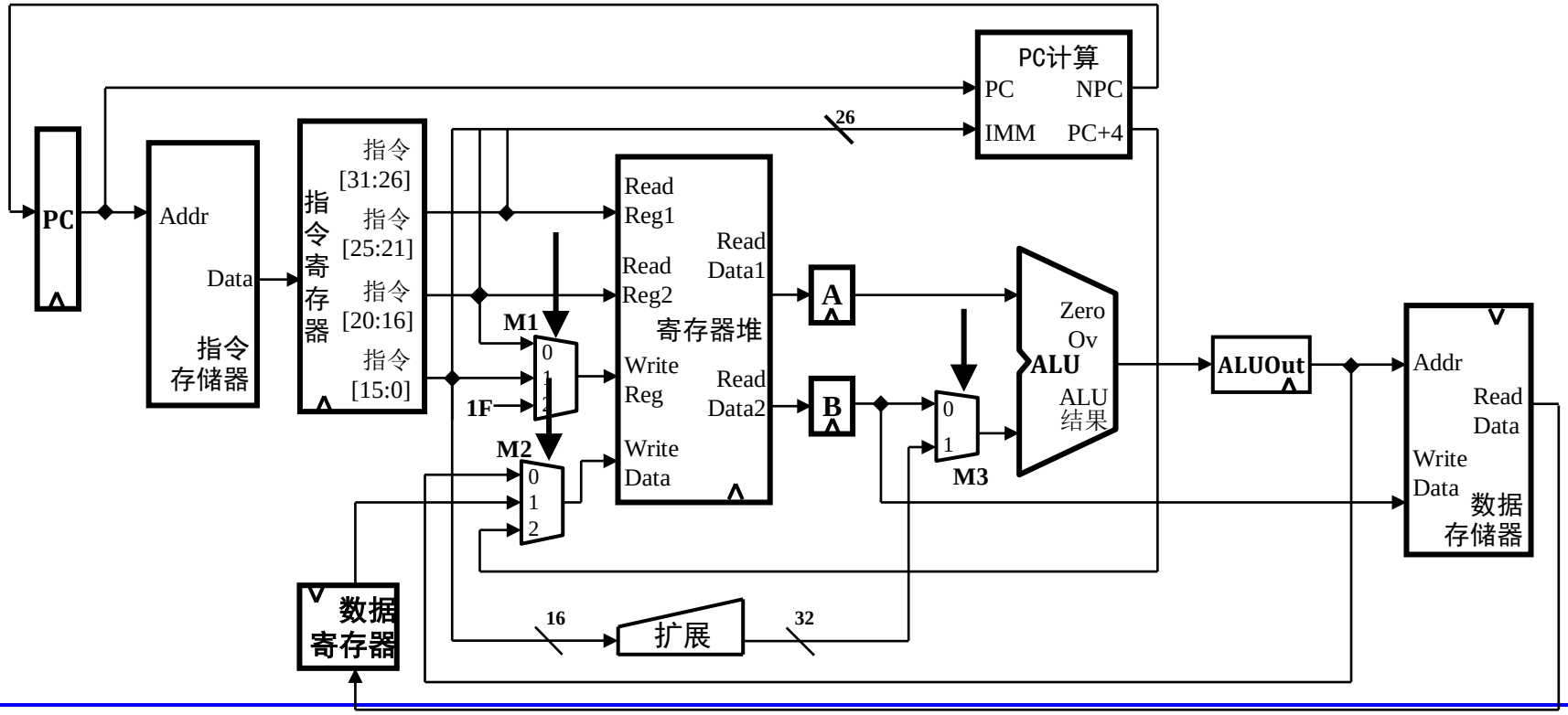
北京航空航天大学计算机学院

提纲

- 数据通路一般性方法
- 状态/控制信号矩阵压缩表达
- 2个数据通路设计的对比

数据通路的一般性方法

- 与单周期没有本质差别
- 增加：IR、A/B、ALUOut、DR
  - 仅仅在原有数据通路中插入寄存器



数据通路的一般性方法

- 扩展了表格
  - 增加IR、A/B、ALUOut、DR
- 分析方法没有任何变化
  - 指令执行的实质路径不因多周期设计而变化

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM	DR
	PC	IMM				WrData	WrReg				1	2			

数据通路的一般性方法

- 数据通路基本特点：大部分部件是固定输入，只有少数为多输入
  - 通用寄存器的目的寄存器：R、I、JAL
  - 通用寄存器的回写数据：运算、load、JAL
  - ALU的2输入

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM	DR
	PC	IMM				WrData	WrReg				1	2			
指令						变	变					变			

示例：LW

- 主要关注：3个
  - WrReg、WrData
  - ALU第2输入

RTL

R[rt]←MEM[R[rs]+sign_ext(imm16)]

PC←PC+4

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM	DR
	PC	IMM				WrData	WrReg				1				
LW	PC	IR[15:0]	NPC.NPC	PC	IM	DR	IR[20:16]	RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut	DM

示例：SW

- SW

RTL

MEM[R[rs]+sign_ext(imm16)] ← R[rt]

PC←PC+4

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM		DR
	PC	IMM				WData	RD				1	2		A	D	
LW	PC	IR[15:0]	NPC.NPC	PC	IM	DR	IR[20:16]	RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut		DM
SW	PC	IR[15:0]	NPC.NPC	PC	IM			RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut	B	

示例：ADDU

- ADDU

RTL

R[rd]←R[rs]+R[rt]
PC←PC+4

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM		DR
	PC	IMM				WData	RD				1	2		A	D	
LW	PC	IR[15:0]	NPC.NPC	PC	IM	DR	IR[20:16]	RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut		DM
SW	PC	IR[15:0]	NPC.NPC	PC	IM			RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut	B	
ADDU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			

示例：SUBU

- SUBU

RTL

R[rd]←R[rs] − R[rt]

PC←PC+4

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM		DR
	PC	IMM				WData	RD				1	2		A	D	
LW	PC	IR[15:0]	NPC.NPC	PC	IM	DR	IR[20:16]	RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut		DM
SW	PC	IR[15:0]	NPC.NPC	PC	IM			RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut	B	
ADDU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
SUBU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			

示例：ORI

- ORI

RTL

R[rd]←R[rs] | zero_extend(imm16)

PC←PC+4

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM		DR
	PC	IMM				WData	RD				1	2		A	D	
LW	PC	IR[15:0]	NPC.NPC	PC	IM	DR	IR[20:16]	RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut		DM
SW	PC	IR[15:0]	NPC.NPC	PC	IM			RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut	B	
ADDU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
SUBU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
ORI	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[20:16]	RF.RD1		IR[15:0]	A	EXT	ALU			

示例：LUI

- 设计方案1
  - 延用R类、I类

RTL

$R[rt] \leftarrow imm16 \,||\, 0^{16}$

$PC \leftarrow PC+4$

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM		DR
	PC	IMM				WData	RD				1	2		A	D	
LW	PC	IR[15:0]	NPC.NPC	PC	IM	DR	IR[20:16]	RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut		DM
SW	PC	IR[15:0]	NPC.NPC	PC	IM			RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut	B	
ADDU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
SUBU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
ORI	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[20:16]	RF.RD1		IR[15:0]	A	EXT	ALU			
LUI	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[20:16]	RF.RD1		IR[15:0]	A	EXT	ALU			

示例：LUI

- 设计方案2
  - 减少部件

RTL

$R[rt] \leftarrow imm16 \,||\, 0^{16}$

$PC \leftarrow PC+4$

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM		DR
	PC	IMM				WData	RD				1	2		A	D	
LW	PC	IR[15:0]	NPC.NPC	PC	IM	DR	IR[20:16]	RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut		DM
SW	PC	IR[15:0]	NPC.NPC	PC	IM			RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut	B	
ADDU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
SUBU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
ORI	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[20:16]	RF.RD1		IR[15:0]	A	EXT	ALU			
LUI	PC	IR[15:0]	NPC.NPC	PC	IM	EXT	IR[20:16]	RF.RD1		IR[15:0]						

示例：BEQ

RTL

PC ← (R[rs]==R[rt]) ? PC+4+sign_ext(imm16) || 00 : PC+4

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM		DR
	PC	IMM				WData	RD				1	2		A	D	
LW	PC	IR[15:0]	NPC.NPC	PC	IM	DR	IR[20:16]	RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut		DM
SW	PC	IR[15:0]	NPC.NPC	PC	IM			RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut	B	
ADDU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
SUBU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
ORI	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[20:16]	RF.RD1		IR[15:0]	A	EXT	ALU			
LUI	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[20:16]	RF.RD1		IR[15:0]	A	EXT	ALU			
BEQ	PC	IR[15:0]	NPC.NPC	PC	IM			RF.RD1	RF.RD2		A	B	ALU			

示例：JAL

RTL

R[31] ← PC + 4

PC ← PC[31:28] || instr_index || 00

	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM		DR
	PC	IMM				WData	RD				1	2		A	D	
LW	PC	IR[15:0]	NPC.NPC	PC	IM	DR	IR[20:16]	RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut		DM
SW	PC	IR[15:0]	NPC.NPC	PC	IM			RF.RD1	RF.RD2	IR[15:0]	A	EXT	ALU	ALUOut	B	
ADDU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
SUBU	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[15:11]	RF.RD1	RF.RD2		A	B	ALU			
ORI	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[20:16]	RF.RD1		IR[15:0]	A	EXT	ALU			
LUI	PC	IR[15:0]	NPC.NPC	PC	IM	ALUOut	IR[20:16]	RF.RD1		IR[15:0]	A	EXT	ALU			
BEQ	PC	IR[15:0]	NPC.NPC	PC	IM			RF.RD1	RF.RD2		A	B	ALU			
JAL	PC	IR[15:0]	NPC.NPC	PC	IM	PC.PC4	0x1F									

提纲

- 数据通路一般性方法
- 状态/控制信号矩阵压缩表达
- 2个数据通路设计的对比

常规表达方式的不足

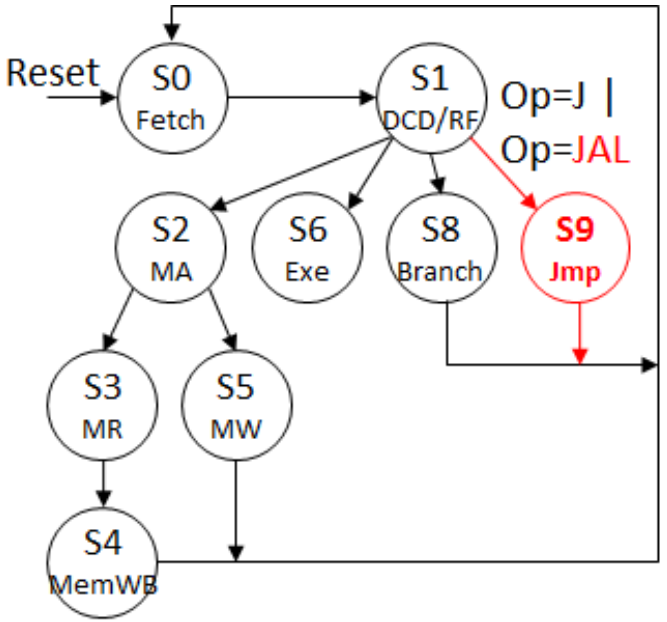
- S1：信号/状态矩阵：每条指令一张表
  - N条指令N张表
- S2：信号真值表矩阵有效表达密度低
  - 大量0元素
  - 我们真正需要却是1元素

	Op	Funct	S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
PCWr			1	0	0	0	0					
NPCOp			PC+4	X	X	X	X					
IRWr			1	0	0	0	0					
GPRWr			0	0	0	0	1					
DMWr			0	0	0	0	0					
ALUOp			X	add	add	add	add					
GPRSel			X	00	00	00	00					
WDSel			X	01	01	01	01					
ExtOp			X	SE	SE	SE	SE					
BSel			X	1	1	1	1					

	Op	Funct	S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
LW	100011		1	0	0	0	0	0	0	0	0	0
SW	101011		1	0	0	0	0	0	0	0	0	0
ADDU	000000	100001	1	0	0	0	0	0	0	0	0	0
SUBU	000000	100011	1	0	0	0	0	0	0	0	0	0
ORI	001101		1	0	0	0	0	0	0	0	0	0
LUI	001111		1	0	0	0	0	0	0	0	0	0
BEQ	000100		1	0	0	0	0	0	0	0	0/1	0
JAL	000011		1	0	0	0	0	0	0	0	0	1

指令分类的启发

- 由于指令分为R、I、J等少数几类，并且指令格式非常简洁，因此：
  - 数据通路：变化点很少
  - 控制逻辑：状态机可以按大类
- 控制信号：大部分取值非常固定
  - 固定的周期，固定的值



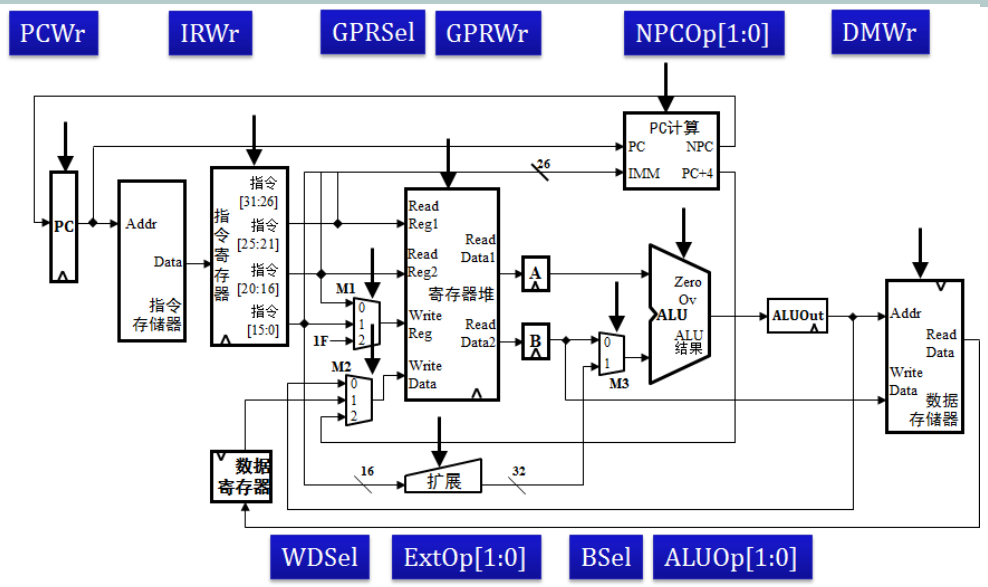
	NPC		PC	IM	IR	RF		A	B	EXT	ALU		ALUOut	DM	DR
	PC	IMM				WrData	WrReg				1	2			
指令						变	变					变			

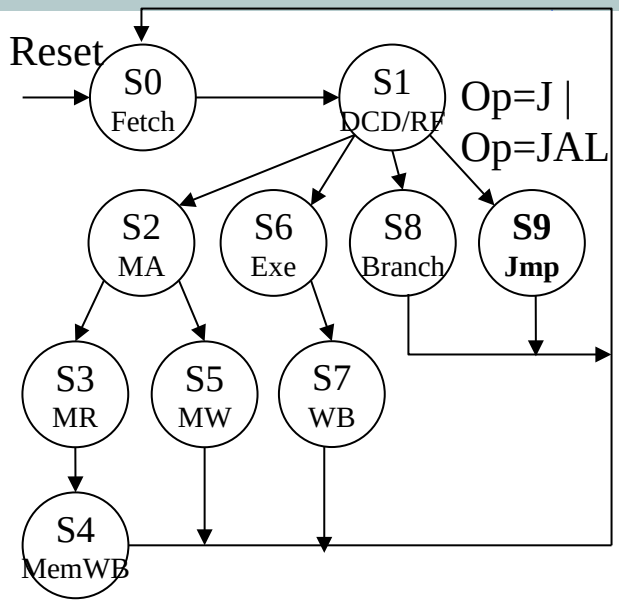
指令分类的启发

- 由于指令分为R、I、J等少数几类，并且指令格式非常简洁，因此：
  - 数据通路：变化点很少
  - 控制逻辑：状态机可以按大类划分

	NPCOp	PCWr	IRWr	WDSel (MUX)	RegDst (MUX)	RegWr	ExtOp	ALUSelB (MUX)	ALUOp	MemWr
ADDU	S0:PC+4	S0	S0	S7:`ALUOut	S7:`RD	S7			S6:ADD	
SUBU	S0:PC+4	S0	S0	S7:`ALUOut	S7:`RD	S7			S6:SUB	
ORI	S0:PC+4	S0	S0	S7:`ALUOut	S7:`RT	S7	S6:`UEXT		S6:OR	
LW	S0:PC+4	S0	S0	S7:`DM	S4:`RT	S4	S2:`SEXT	S2:2	S2:ADD	
SW	S0:PC+4	S0	S0					S2:2	S2:ADD	S5
BEQ	S0:PC+4 S8:BEQ	S0 S8·Zero	S0						S8:SUB	
JAL	S0:PC+4 S9:JMP	S0 S9	S0	S9:NPC	S9:`1F	S9		S1:3	S9:ADD	

你发现了什么？





	NPCOp	PCWr	IRWr	WDSel (MUX)	GPRSel (MUX)	GPRWr	ExtOp	ALUSelB (MUX)	ALUOp	DMWr
ADDU	S0:PC+4	S0	S0	S7:`ALUOut	S7:`RD	S7			S6:ADD	
SUBU	S0:PC+4	S0	S0	S7:`ALUOut	S7:`RD	S7			S6:SUB	
ORI										
LW										
SW										5
BEQ	S0:PC+4 S8:BEQ	S0 S8·Zero	S0						S8:SUB	
JAL	S0:PC+4 S9:JMP	S0 S9	S0	S9:NPC	S9:`1F	S9		S1:3	S9:ADD	

同时给出控制信号的{时间，有效值}

- 不再关心无效值

怎么写表达式？

PCWr = S0 + beq·S8·zero + jal·S9

PCWr = (lw+sw+addu+subu+ori+lui+beq+jal)·S0 +

beq·S8·zero +

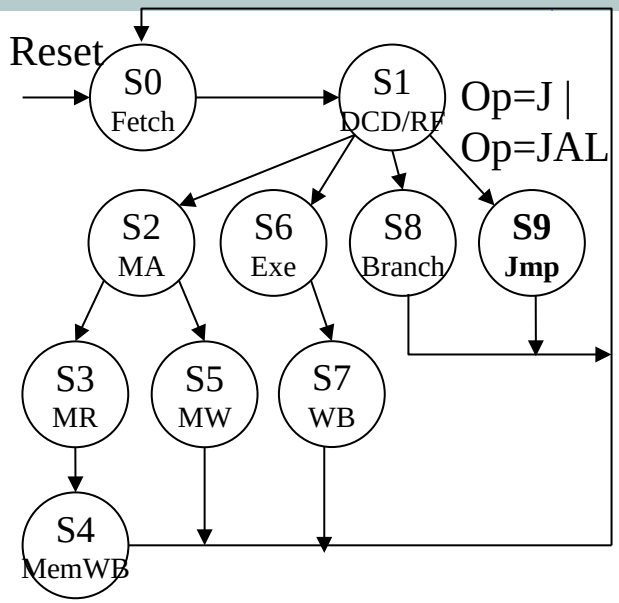
jal·s9

昨天的PPT

	NPCOp	PCWr	IRWr	WDSel (MUX)	RegDst (MUX)	RegWr	ExtOp	ALUSelB (MUX)	ALUOp	MemWr
ADDU	S0:PC+4	S0	S0	S7:`ALUOut	S7:`RD	S7			S6:ADD	
SUBU	S0:PC+4	S0	S0	S7:`ALUOut	S7:`RD	S7			S6:SUB	
ORI	S0:PC+4	S0	S0	S7:`ALUOut	S7:`RT	S7	S6:`UEXT		S6:OR	
LW	S0:PC+4	S0	S0	S7:`DM	S4:`RT	S4	S2:`SEXT	S2:2	S2:ADD	
SW	S0:PC+4	S0	S0					S2:2	S2:ADD	S5
BEQ	S0:PC+4 S8:BEQ	S0 S8·Zero	S0						S8:SUB	
JAL	S0:PC+4 S9:JMP	S0 S9	S0	S9:NPC	S9:`1F	S9		S1:3	S9:ADD	

状态机的设计思路

- 指令划分为若干大类
- 状态机按设计大类
  - FETCH→DCD/RD是共性基础
  - 分支均在DCD/RD处判定
- 若某条指令归入任一分类都感觉不尽合理
  - 方法：设置一个新的分支

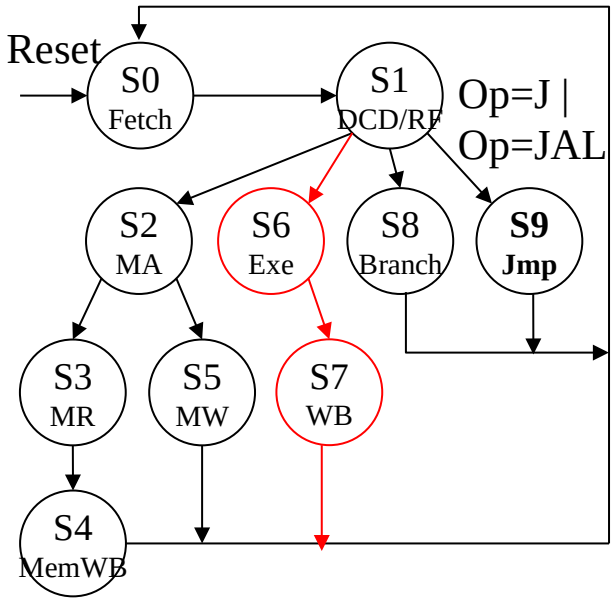


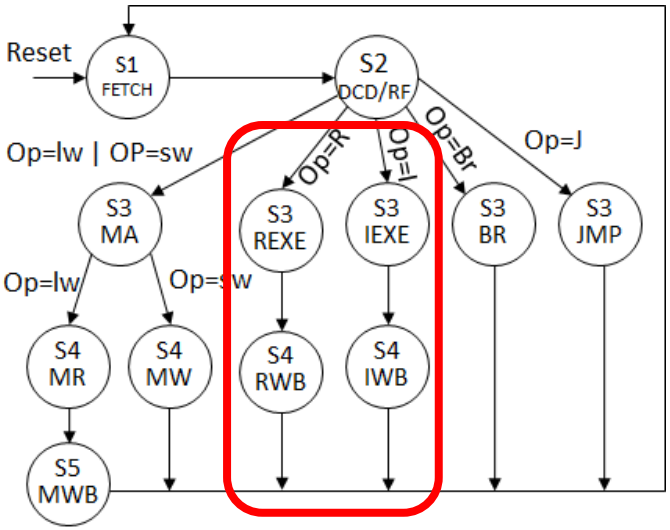
按照指令大类设计的意义

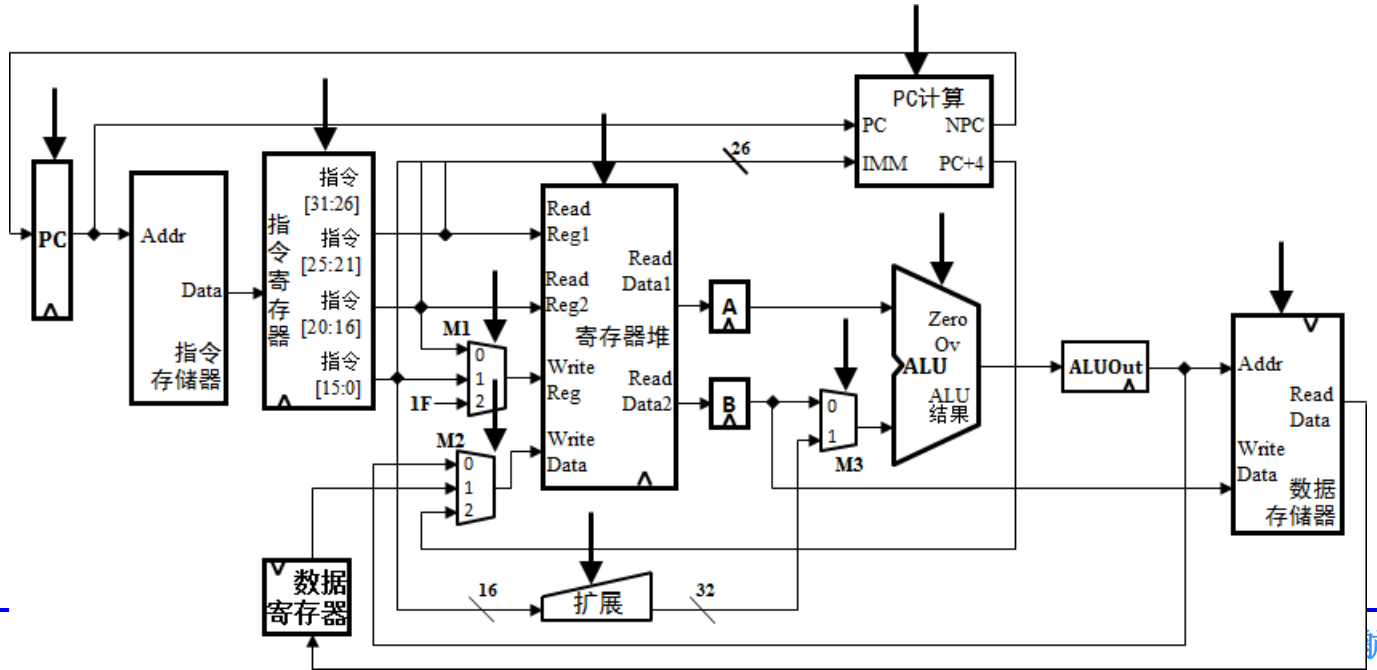
- Q：R类的主要区别是什么？
  - A：ALU应该执行什么操作！
- Q：怎么解决？
  - A：根据指令设置正确的ALUOp即可
- 从状态机的角度，不需要关心到R类内部细节
  - 只需要关注R类的执行策略和路径
  - 路径某个节点发生了什么，是低一个层次问题
    - 在指令/信号矩阵解决

有没有标准的状态机？

- A：没有！
- 均可行
  - 差别很细微







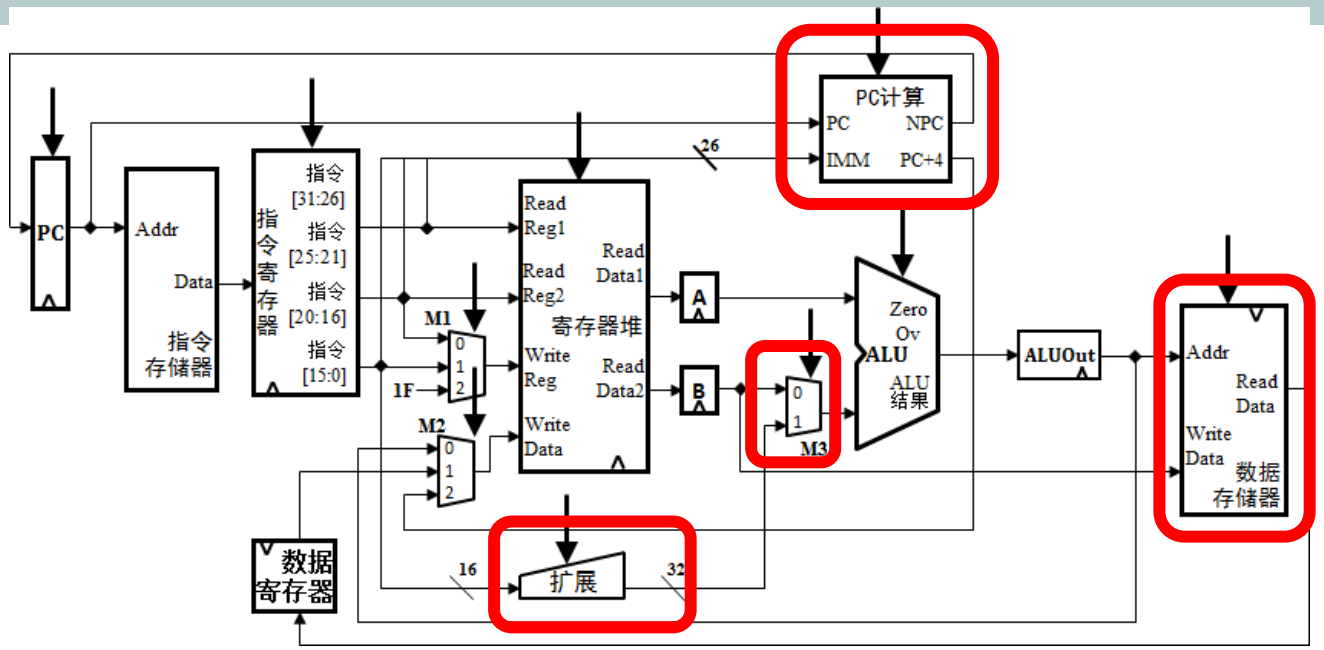
提纲

- 数据通路一般性方法
- 状态/控制信号矩阵压缩表达
- 2个数据通路设计的对比

PPT数据通路 vs 教科书数据通路

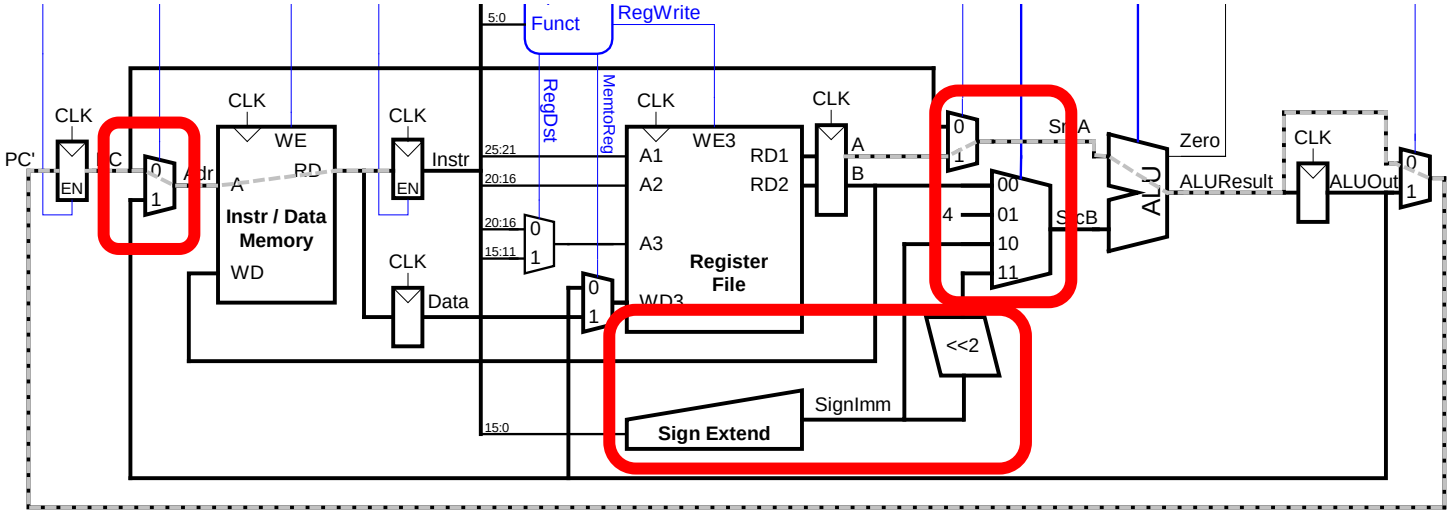
PPT

数据通路



教科书

数据通路



PPT数据通路 vs 教科书数据通路

- PPT：保留NPC，与单周期、流水线一致
  - 整个设计更加简洁，数据通路的结构化特征更好
    - 设计思想：高内聚，低耦合
  - 所有的与地址产生的都放在NPC中（包括今后的启动地址和异常地址）
  - IM地址端，ALU的A端：没有MUX
    - 教科书：为了J类指令
- 教科书：虽然可节省一点逻辑，但清晰度不好
  - 只有1个ALU，节省了NPC里的加法器
  - 状态机设计更加复杂，更有利于学习“状态机”

谢谢大家！